UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

# REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION
UNCLASSIFIED

1b. RESTRICTIVE MARKINGS

AD-A215 083

DTIC
ELECTE
DEC 01 1989
D
CS

3. DISTRIBUTION/AVAILABILITY OF REPORT
Approved for public release; distribution unlimited.

(2)

5. MONITORING ORGANIZATION REPORT NUMBER(S)
AFOSR-TR-89-1488

| 6a. NAME OF PERFORMING ORGANIZATION | 6b. OFFICE SYMBOL (If applicable) | 7a. NAME OF MONITORING ORGANIZATION |
|---|---|---|
| University of Arizona | | Air Force Office of Scientific Research |

| 6c. ADDRESS (City, State, and ZIP Code) | 7b. ADDRESS (City, State, and ZIP Code) |
|---|---|
| Tucson, AZ 85721 | Building 410<br>Bolling AFB, DC 20332-6448 |

| 8a. NAME OF FUNDING/SPONSORING ORGANIZATION | 8b. OFFICE SYMBOL (If applicable) | 9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER |
|---|---|---|
| AFOSR | NM | AFOSR-88-0147 |

| 8c. ADDRESS (City, State, and ZIP Code) | 10. SOURCE OF FUNDING NUMBERS | | | |
|---|---|---|---|---|
| | PROGRAM ELEMENT NO. | PROJECT NO. | TASK NO | WORK UNIT ACCESSION NO. |
| Building 410<br>Bolling AFB, DC 20332-6448 | 61102F | 2304 | A2 | |

11. TITLE (Include Security Classification)

THE SAGUARO DISTRIBUTED OPERATING SYSTEM

12. PERSONAL AUTHOR(S)
Gregory R. Andrews

| 13a. TYPE OF REPORT | 13b. TIME COVERED | 14. DATE OF REPORT (Year, Month, Day) | 15. PAGE COUNT |
|---|---|---|---|
| FINAL | FROM 01 Mar 88 TO 28 Feb 89 | | |

16. SUPPLEMENTARY NOTATION

| 17. COSATI CODES | | | 18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number) |
|---|---|---|---|
| FIELD | GROUP | SUB-GROUP | |
| | | | |
| | | | |

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

The progress achieved over the final year of the Saguaro distributed operating system proj is presented. The primary achievements were in related research, including SR distributed programming language, the MLP system for constructing distributed mixed-language programs, the Psync interprocess communication mechanism, a configurable operating system kernal cal the x-kernal, and the development of language mechanisms for performing failure handling in distributed programming languages.

| 20. DISTRIBUTION/AVAILABILITY OF ABSTRACT | 21. ABSTRACT SECURITY CLASSIFICATION |
|---|---|
| ☐ UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT. ☐ DTIC USERS | UNCLASSIFIED |

| 22a. NAME OF RESPONSIBLE INDIVIDUAL | 22b. TELEPHONE (Include Area Code) | 22c. OFFICE SYMBOL |
|---|---|---|
| DR ABRAHAM WAKSMAN | (202) 767-5027 | NM |

DD Form 1473, JUN 86 Previous editions are obsolete. SECURITY CLASSIFICATION OF THIS PAGE



UNCLASSIFIED

# Final Technical Report

## The Saguaro Distributed Operating System



Grant Number: AFOSR-88-0147

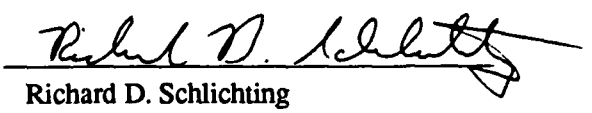
Grant Duration: March 11, 1988 - February 28, 1989

Awarded to: The University of Arizona
Tucson, Arizona 85721

Principal Investigators: Gregory R. Andrews
Richard D. Schlichting
Department of Computer Science

AFOSR Program Manager: Dr. Abraham Waksman
Directorate on Mathematical and Information Sciences

Gregory R. Andrews
May 24, 1989



Richard D. Schlichting
May 24, 1989

**Summary**

The progress achieved over the final year of the Saguaro distributed operating system project is presented. The primary achievements were in related research, including the SR distributed programming language, the MLP system for constructing distributed mixed-language programs, the Psync interprocess communication mechanism, a configurable operating system kernel called the *x*-kernel, and the development of language mechanisms for performing failure handling in distributed programming languages.

## Research Accomplishments

In the past year, we have made significant progress on work related to the Saguaro distributed operating system. This projects include the SR distributed programming language, the MLP system for constructing distributed mixed-language programs, the Psync interprocess communication mechanism, a configurable operating system kernel called the *x*-kernel, and the design of language mechanisms for handling failures in distributed programs.

The first project is involved with further development of the SR distributed programming language, the language used to implement Saguaro. Over the past several years, a large amount of effort was expended revising SR and implementing a compiler for the new version. The main language constructs are still resources and operations: resources encapsulate processes and variables they share, operations provide the primary mechanism for process interaction. One way in which SR has changed is that both resources and processes are now created dynamically. Another change is that the mechanisms for operation invocation—**call** and **send**—and operation implementation—**proc** and **in**—have been extended and integrated. Consequently, all of local and remote procedure call, rendezvous, dynamic process creation, asynchronous message passing, multicast, and semaphores are supported. We have found this flexibility to be very useful for distributed programming. The language has also been refined in numerous additional ways to provide additional flexibility. Moreover, by basing SR on a small number of well-integrated concepts, the language is also relatively simple and has a reasonably efficient implementation.

The compiler and runtime support currently execute on top of UNIX; they have been in use since November 1985. Since that time, numerous enhancements to the language have been made, mainly in response to comments from users. SR now runs on both Vaxes and Suns. The implementation was released into the public domain in 1988 and there are currently over 50 sites. Two papers on SR-related topics authored by Stella Atkins, who was a visiting professor at Arizona during Spring 1986, recently appeared [1, 2].

A second Saguaro-related project is the MLP system for constructing distributed, mixed language programs. Specifically, the system is a simple RPC (remote procedure call) system that allows each procedure in a sequential program to be written in a different programming language and executed on any machine in a network. MLP is based on a type scheme called UTS (the *Universal Type System*), which is composed of a data representation standard and a type expression language. The type expression language is used as an interface definition language in a manner similar to the use of Courier in Xerox systems; the data representation standard specifies the format of data transferred as arguments or results between procedures in a machine- and language-independent fashion. The initial version of the system supports C, Pascal, and Icon; it has been in use since November 1986 on an interconnected collection of Vaxes and Suns running Berkeley UNIX. Work has currently underway on a second version of the system.

A description of the implementation of MLP and our experience in using the system appears in [3]. The way in which the object-oriented distributed programming language Emerald is being integrated into the second version of the system appears in [4]. A dissertation that describes how UTS has been used in both MLP and Saguaro is also nearing completion [5].

Another line of research has involved two separate projects investigating aspects of interprocess communication. The first is Psync, an interprocess communication mechanism that supports a new abstraction through which a group of processes can exchange messages. The novel aspect of Psync is that it explicitly embeds timing information drawn from the distributed computation's *logical clock*. We have recently implemented and experimented with a prototype version of Psync. The prototype demonstrates that explicitly preserving this partial ordering in the communications subsystem can be implemented on an unreliable communications network at little cost. Also, the abstraction provided by Psync is general enough to support efficient and elegant implementations of a wide spectrum of communication paradigms.

The second project is the *x*-kernel, a configurable operating system kernel designed to support experimentation in interprocess communication and distributed programming. The *x*-kernel's underlying architecture provides a rich set of abstractions that are used to construct and compose communication protocols. The architecture is interesting because the these abstractions are both general enough to accommodate a wide range of protocols and efficient enough to provide a useful testbed in which protocol performance can be accurately measured.

A comprehensive paper describing Psync and its implementation has been accepted for publication [6]. The design of the *x*-kernel is documented in [7], and a paper outlining the system and its implementation is pending [8]. Two other papers relating aspects of the *x*-kernel have also been written [9, 10].

The final line of investigation has been developing language mechanisms for handling failures. One property that makes failures difficult to handle in such systems is that they can occur concurrently with other system events. We have been investigating an approach for writing fault-tolerant distributed programs that can cope with such asynchrony in a systematic manner. The basic idea is to treat failures as just another class of events that are handled similarly to normal system events. Linguistic constructs that can be added to distributed programming languages with minimal impact are then proposed to handle such failure events. To make our ideas precise, we are using the SR distributed programming language as a basis for incorporating these constructs. A paper outlining our approach is scheduled to appear in August [11].

Finally, a paper appeared that surveys several of these projects in distributed languages and systems, and offers observations based on the experience gained during their design, implementation, and use [12]. The relevant projects are the SR distributed programming language, the Saguaro distributed operating system, the MLP system for constructing distributed mixed-language programs, the object-based distributed programming language Emerald, and the Psync interprocess communication mechanism. The observations address the experimentation process itself as well as the design of distributed software.

## References


[1] Atkins, M.S and Olsson, R.A. Performance of multi-tasking and synchronization mechanisms in the programming language SR. *Software—Practice and Experience 18,9* (Sept. 1988), 879-895.

[2] Atkins, M.S. Experiments in SR with different upcall program structures. *ACM Trans. on Computer Systems 6,4* (Nov. 1988), 365-392.

[3] Hayes, R., Manweiler, S.W., and Schlichting, R.D. A simple system for constructing distributed, mixed language programs. *Software—Practice and Experience 18,7* (July 1988), 641-660.

[4] Hayes, R., Hutchinson, N.C., and Schlichting, R.D. Integrating Emerald into a system for mixed-language programming. Submitted for publication, October 1988.

[5] Hayes, Roger. UTS: A Type System for Facilitating Data Communication. Ph.D. Dissertation, Department of Computer Science, University of Arizona, Aug. 1989 (expected).

[6] Peterson, L.L., Buchholz, N., and Schlichting, R.D. Preserving and using context information in interprocess communication. *ACM Trans. on Computer Systems* (1989), to appear.

[7] Hutchinson, N.C., and Peterson, L.L. Design of the *x*-kernel. *Proc. ACM SIGCOMM '88*, Stanford, CA (Aug. 1988), 65-75.

[8] Hutchinson, N.C., and Peterson, L.L. Implementing protocols in the *x*-kernel. Submitted for publication, January 1989.

[9] Hutchinson, N.C., Mishra, S., Peterson, L.L., and Thomas, V. Tools for implementing network protocols. *Software—Practice and Experience* (1989), to appear.

[10] Abbott, M.B., Hutchinson, N.C., O'Malley, S., and Peterson, L.L. RPC in the *x*-kernel: Evaluating new design techniques. *Proc. 12th ACM Symp. on Operating Sys. Principles*, Litchfield Park, AZ (Dec. 1989), to appear.

[11] Schlichting, R.D., Cristian, F., and Purdin, T. A linguistic approach to failure handling in distributed systems. *IFIP International Working Conference on Dependable Computing for Critical Applications*, Santa Barbara, CA (August 1989), to appear.

[12] Schlichting, R.D., Andrews, G.R., Hutchinson, N.C., Olsson, R.A., and Peterson, L.L. Observations on building distributed languages and systems. *Experiences with Distributed Systems*, (J. Nehmer, Ed.), *Lecture Notes in Computer Science*, Vol. 309, Springer-Verlag, New York, 1988, 271-291.

## Participating Professionals

Gregory R. Andrews, Professor and Department Head, Dept. of Computer Science, University of Arizona

Richard D. Schlichting, Associate Professor, Dept. of Computer Science, University of Arizona

Norman C. Hutchinson, Assistant Professor, Dept. of Computer Science, University of Arizona

Larry L. Peterson, Assistant Professor, Dept. of Computer Science, University of Arizona

Ronald A. Olsson, Assistant Professor, Division of Computer Science, University of California, Davis (Ph.D. earned at Arizona, 1986)

Titus Purdin, Assistant Professor, Dept. of Computer Science, Colorado State University (Ph.D. earned at Arizona, 1987)

Stella Atkins, Assistant Professor, Dept. of Computer Science, Simon Fraser University (visiting Assistant Professor, 1986)

Steve W. Manweiler, Member, Technical Staff, Hewlett-Packard Corp., Ft. Collins, Colorado (M.S. earned at Arizona, 1986)

Mark Abbott, Ph.D. student, Dept. of Computer Science, University of Arizona

Nick Buchholz, Ph.D. student, Dept. of Computer Science, University of Arizona

Mike Coffin, Ph.D. student, Dept. of Computer Science, University of Arizona

Irv Elshoff, Ph.D. student, Dept. of Computer Science, University of Arizona

Roger Hayes, Ph.D. student, Dept. of Computer Science, University of Arizona

Shivakant Mishra, Ph.D. student, Dept. of Computer Science, University of Arizona

Sean O'Malley, Ph.D. student, Dept. of Computer Science, University of Arizona

Vicraj Thomas, Ph.D. student, Dept. of Computer Science, University of Arizona